UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/567,851	10/06/2006	Jouni Kytomaa	39700-638N01US/NC40070US	7339

64046 7590 04/27/2011

MINTZ, LEVIN, COHN, FERRIS, GLOVSKY AND POPEO, P.C
ONE FINANCIAL CENTER
BOSTON, MA 02111

EXAMINER	
MITCHELL, DANIEL D	

ART UNIT	PAPER NUMBER
2477	

MAIL DATE	DELIVERY MODE
04/27/2011	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/567,851	KYTOMAA ET AL.	
	Examiner	Art Unit	
	DANIEL MITCHELL	2477	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 January 2011*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17 and 35-46* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17 and 35-46* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 1/13/2011 has been entered.

Response to Amendment

2. Applicant's amendment filed on 1/13/2011 has been entered. Claims 1, 2, 35, and 46 have been amended. Claims 17-34 are canceled. Claims 1-17 and 35-46 are still pending in this application, with claims 1, 2, 35, and 46 being independent.

Response to Arguments

3. Applicant's arguments with respect to claims 1-46 have been considered but are moot in view of the new ground(s) of rejection.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the

invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-12, 14-17, and 35-46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Oner (US Publication No. 2005/0078694 A1), in view of Shimazu et al. (US Publication No. 2004/0125815 A1), hereinafter referred as Shimazu in further view of Cherukuri et al. (US Patent No. 6,976,096 B1), hereinafter referred as Cherukuri.

Regarding claim 1, Oner teaches a method comprising: allocating each received packet, where priority information is in the received packet **(par. 80 suggests that each received packet includes priority information within the header of the received packet)**, to at least one arrival queue of a plurality of arrival queues **(par. 50 teaches placing the packets into a receive queue; fig. 3 teaches a plurality of arrival queues 338)**, wherein each received packet comprises an internet protocol packet **(par. 27 teaches receiving an IP packet)**;

scheduling, by a scheduler coupled to the at least one arrival queue, packets from the arrival queue to at least one transfer queue **(fig. 3 teaches the scheduler coupled to the at least one arrival queue (receiver buffer - 338); par. 51-53 teaches scheduling, by a scheduler, the packets from the receive buffers to the transfer queues of the packet manager 320; par. 85 further teaches the plurality of data buffers (transfer queues) of the packet manager 320)**;

responsive to transfer of a packet to a transfer queue, generating an interrupt **(par. 77 teaches generating an interrupt for each transfer queue of the packet manager 320)**;

responsive to receipt of an interrupt, allocating the packet from said transfer queue to the memory of the processor **(par. 58 suggests the packet is sent to the processor memory 340 upon the interrupt)**;

placing the packet in the processor memory 340 if said memory is not full **(par. 55 teaches transferring the data from the transfer queues of the packet manager 320 to the processor memory 340 when space is available)**, wherein the at least one transfer queue and the processor memory do not drop packets **(par. 55 suggests packet are not dropped because packets are only transferred as space is available in the memories)**, and wherein the scheduler inhibits placement of the packet when the processor memory is full to prevent dropping the packet otherwise dropping said packet **(par. 55 teaches packets are only placed in queues when there is space available)**; and

scheduling packets from the processor memory 340 to be processed **(par. 55 teaches storing the packets in the memory; par. 58 further teaches synchronizing (scheduling) the data for processing by the CPUs)**, wherein the at least one arrival queue **(receive buffers 338)**, the at least one transfer queue **(queues of packet manager 320)**, and processor memory **(processor memory 340)** are separate queues **(fig. 3 teaches separate storages)**, wherein the scheduler includes a first quantity N of inputs each corresponding to the at

least one arrival queue, the scheduler further including a second quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N **(fig. 3 teaches a scheduler with at least one input and at least one output, where the input is coupled to the input receive buffers and the output is couple to the transfer queues of the unit 320)**.

Oner suggests in par. 80 that each incoming packet includes various packet attribute values (priority information) in the header of the packet however, Oner does not expressly disclose allocating each received packet, based on priority information in the received packet, to at least one arrival queue of a plurality of arrival queues, wherein each of the plurality of arrival queues handles packets based on a traffic class associated with a priority; and placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet; and the processor memory includes a plurality of processor queues.

Shimazu teaches in par. 190 allocating each received packet, based on priority information in the received packet, to at least one arrival queue of a plurality of arrival queues, wherein each of the plurality of arrival queues handles packets based on a traffic class associated with a priority **(par. 190 teaches allocating each received packet to one of a plurality of queues based on the priority of the queue).**

Shimazu teaches placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet **(par. 190 teaches allocating packets to various queues; par. 199-200 teaches if the queues are full the packets that are to be placed into each queue will be discarded).**

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner to include prioritizing packets into arrival queues. One would be motivated as such in order to provide different service to the received packets.

However Oner and Shimazu do not expressly disclose the processor memory includes a plurality of processor queues.

Cherukuri teaches a plurality of processor queues **(330-336)** in fig. 3, col. 7 lines 19-22 for receiving packets from transfer queues for future processing.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner and Shimazu to include a plurality of processor queues in the processor memory. One would be motivated as such in order to process packets in order of priority.

Regarding claim 2, Oner teaches a method comprising: receiving a plurality of packets at one or more of a plurality of network devices coupled via one or more buses to at least one of a processor and a memory **(par. 19 teaches network devices, processors and memory coupled by buses)**,

wherein the packet comprises an internet protocol packet **(par. 27 teaches receiving an IP packet at an interface device)**;

allocating each received packet to at least one arrival queue **(par. 50 teaches placing the packets into a receive queue; fig. 3 teaches a plurality of arrival queues 338)**,

wherein the one or more of the plurality of network devices comprises a scheduler **(scheduler)**, the at least one arrival queue **(buffer 338)**, and at least one transfer queue **(transfer queues of the packet manger) (fig. 3 teaches the structure)**,

wherein the at least one of the processor and the memory further comprises at processor memory **(fig. 3 teaches a memory 340 for storing data for the processor)**,

wherein the scheduler is further coupled to the at least one transfer queue **(fig. 3 teaches the scheduler is coupled to the transfer queues of the packet manager 320)**;

scheduling, by the scheduler coupled to the at least one arrival queue, each packet from the arrival queue to the at least one transfer queue **(fig. 3 teaches the scheduler coupled to the at least one arrival queue (receiver buffer - 338); par. 51-53 teaches scheduling, by a scheduler, the packets from the receive buffers; par. 85 further teaches the plurality of data buffers (transfer queues) of the packet manager 320)**;

responsive to transfer of a packet to the at least one transfer queue, generating an interrupt **(par. 77 teaches generating an interrupt for each transfer queue of the packet manager)**;

responsive to receipt of the interrupt, allocating the packet from the at least one transfer queue to the processor memory **(par. 58 suggests the packet is sent to the processor memory)**;

placing the packet in the processor memory if said memory is not full **(par. 55 teaches transferring the data from the queues of the packet manager 320 to the processor queues (memory 340) when space is available)**, wherein the at least one transfer queue and the at least one processor memory do not drop packets **(par. 55 suggests packet are not dropped because packets are only transferred as space is available in the memories)**, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet otherwise dropping said packet **(par. 55 teaches packets are only placed in queues when there is space available)**; and

scheduling packets from the at least one of the plurality of processor queues to be processed **(par. 55 teaches storing the packets in the memory; par. 58 further teaches synchronizing (scheduling) the data for processing by the CPUs)**, wherein the at least one arrival queue **(receiver buffers)**, the at least one transfer queue **(queue of the packet manager)**, and the plurality of

processor memory **(memory 340)** are separate queues **(fig. 3 teaches separate storages)**.

Oner suggests in par. 80 that each incoming packet includes various packet attribute values (priority information) in the header of the packet however, Oner does not expressly disclose and placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet; and the processor memory includes a plurality of processor queues.

Shimazu teaches placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet **(par. 190 teaches allocating packets to various queues; par. 199-200 teaches if the queues are full the packets that are to be placed into each queue will be discarded).**

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner to include prioritizing packets into arrival queues. One would be motivated as such in order to provide different service to the received packets.

However Oner and Shimazu do not expressly disclose the processor memory includes a plurality of processor queues.

Cherukuri teaches a plurality of processor queues **(330-336)** in fig. 3, col. 7 lines 19-22 for receiving packets from transfer queues for future processing.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner and Shimazu to include

a plurality of processor queues in the processor memory. One would be motivated as such in order to process packets in order of priority

Regarding claim 3, Oner teaches wherein at least one device has a plurality of arrival queues **(fig. 3 teaches a system 300 including a plurality of receive buffers)**.

Regarding claim 4, Oner, Shimazu, and Cherukuri teach a method as the parent claim.

Oner suggests in par. 80 that each incoming packet includes various packet attribute values (priority information) in the header of the packet however, Oner does not expressly disclose wherein each arrival queue is associated with a traffic class, each packet being allocated to the at least one queue in accordance with the traffic class of each packet.

Shimazu wherein each arrival queue is associated with a traffic class, each packet being allocated to the at least one queue in accordance with the traffic class of each packet **(par. 190 teaches each input queue is associated with a priority where each packet is allocated to the queue based on priority).**

See similar motivation as claim 2.

Regarding claim 5, Oner, Shimazu, and Cherukuri teach a method as the parent claim. Further Oner teaches wherein the traffic class is priority information embedded in the each packet **(par. 80 teaches each packet having packet attribute information within the header).**

Regarding claim 6, Oner teaches wherein at least one device comprises a plurality of transfer queues **(fig. 3, par. 85 teaches a plurality of transfer queues of the packet manager).**

Regarding claim 7, Oner teaches wherein the number of transfer queues is less than the number of arrival queues **(fig. 3 suggest the number of transfer queues is less than the number of arrival queues)**.

Regarding claim 8, Oner, Shimazu, and Cherukuri teach a method as the parent claim.

However Oner does not expressly disclose wherein the scheduling of packets from the arrival queue to the transfer queue is dependent upon one or more of: the traffic profile.

Shimazu teaches wherein the scheduling of packets from the arrival queue to the transfer queue is dependent upon one or more of: the traffic profile **(par. 193 teaches scheduling packets from the arrival queue is based on the traffic profile (priority)).**

(par. 50 teaches placing the packets into a receive queue; fig. 3 teaches a plurality of arrival queues 338).

See similar motivation as claim 2.

Regarding claim 9, Oner teaches wherein the transfer queue comprises a device level transfer queue and a processor level transfer queue **(par. 85 teaches the two level of transfer queues; fig. 3 teaches device level transfer queues and processor level transfer queues)**, wherein the device level transfer queue receives packets from the arrival queue, and the processor level transfer queue receives packets from the device level transfer queue **(fig. 3 teaches device level transfer queues and processor level transfer queues within element 320 where the device level transfer queue receives packets from the arrival queues 338)**.

Regarding claim 10, Oner teaches wherein packets are transferred to the processor level transfer queue from the device level transfer queue whenever there is space in the processor level transfer queue **(par. 55 teaches allocating data from a transfer queue to a processor queue when space is available).**

Regarding claim 11, Oner teaches a method according to claim 10, wherein packets are never dropped from the transfer queue **(par. 53 teaches**

transfer queues that buffer data; par. 53 does not teach of dropping packets from the transfer queue).

Regarding claim 12, Oner, Shimazu, and Cherukuri teach a method as the parent claim.

However Oner and Shimazu do not expressly disclose queues are associated with different priorities.

Cherukuri teaches wherein the processor queues are associated with different priorities **(col. 7 lines 9-12 teaches the queues are associated with different priorities).**

See similar motivation as claim 2.

Regarding claim 14, Oner teaches wherein responsive to receipt of the interrupt, a packet is removed from a transfer queue and classified. **(par. 29 teaches transferring a packet for processing responsive to an interrupt to a queue and classifies the packet to a queue).**

Regarding claim 15, Oner, Shimazu, and Cherukuri teach a method as the parent claim.

However Oner does not expressly disclose wherein the classification is based on a determination of priority.

Shimazu teaches wherein the classification is based on a determination of priority **(par. 190 classifying packets to a queue based on priority).**

See similar motivation as claim 2.

Regarding claim 16, Oner, Shimazu, and Cherukuri teach a method as the parent claim.

However Oner and Shimazu do not expressly disclose wherein the packet is allocated to a processor queue in accordance with a classification of the packet.

Cherukuri teaches wherein the packet is allocated to a processor queue in accordance with a classification of the packet **(col. 7 lines 9-12 teaches allocating a packet to a queue based on the classification of priority).**

See similar motivation as claim 2.

Regarding claim 17, Oner teaches wherein the packet is placed in the allocated processor queue if said queue is not full, otherwise the packet is dropped **(par. 55 teaches placing data in memory when there is space available).**

Regarding claim 35, Oner teaches An apparatus comprising: a processor configured to allocate a received packet to at least one arrival queue **(par. 50 teaches placing the packets into a receive queue; fig. 3 teaches a plurality**

of arrival queues 338), wherein the received packet comprises an internet protocol packet **(par. 27 teaches receiving an IP packet)**;

wherein the processor is configured to schedule packets from the arrival queue to at least one transfer queue, wherein the processor is responsive to transfer of a packet to a transfer queue **(fig. 3 teaches the scheduler being couple to the at least one arrival queue (receiver buffer - 338); par. 51-53 teaches scheduling, by a scheduler, the packets from the receive buffers to the transfer queues; par. 85 further teaches the plurality of data buffers (transfer queues) of the packet manager 320)**;

configured to generate an interrupt, wherein the processor is responsive to receipt of an interrupt **(par. 77 teaches generating an interrupt for each transfer queue of the packet manager)**,

configured to allocate the packet from said transfer queue to processor memory **(par. 58 suggests the packet is sent to the processor memory 340)**,

wherein the processor is configured to place the packet in the allocated processor memory if said processor memory is not full **(par. 55 teaches transferring the data from the queues of the packet manager 320 to the processor queues (memory 340) when space is available)**, wherein the at least one transfer queue and the memory do not drop packets **(par. 55 suggests packet are not dropped because packets are only transferred as space is available in the memories)**, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet **(par. 55**

teaches packets are only placed in queues when there is space available), and

wherein the processor is configured to schedule packets from the processor memory to be processed **(par. 55 teaches storing the packets in the memory; par. 58 further teaches synchronizing (scheduling) the data for processing by the CPUs)**

wherein the at least one arrival queue **(receive queue)**, the at least one transfer queue **(transfer queues of the packet manager)**, and the processor memory **(memory 340)** are separate queues, wherein the scheduler includes a first quantity N of inputs each corresponding to the at least one arrival queue, the scheduler further including a second quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N **(fig. 3 teaches a scheduler with at least one input and at least one output, where the input is coupled to the input receive buffers and the output is couple to the transfer queues of the unit 320)**.

Oner suggests in par. 80 that each incoming packet includes various packet attribute values (priority information) in the header of the packet however, Oner does not expressly disclose placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet; and the processor memory includes a plurality of processor queues.

Shimazu teaches placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet **(par. 190 teaches**

allocating packets to various queues based on priority; par. 199-200 teaches if the queues are full the packets that are to be placed into each queue will be discarded).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner to include prioritizing packets into arrival queues. One would be motivated as such in order to provide different service to the received packets.

However Oner and Shimazu do not expressly disclose the processor memory includes a plurality of processor queues.

Cherukuri teaches a plurality of processor queues **(330-336)** in fig. 3, col. 7 lines 19-22 for receiving packets from transfer queues for future processing.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner and Shimazu to include a plurality of processor queues in the processor memory. One would be motivated as such in order to process packets in order of priority.

Regarding claim 36, Oner teaches further comprising a plurality of arrival queues **(fig. 3 teaches a plurality of arrival queues)**.

Regarding claim 37, Oner, Shimazu and Cherukuri teach an apparatus as the parent claim.

However Oner does not expressly disclose wherein each arrival queue is associated with a traffic class, each packet being allocated to at least one queue by the processor in accordance with the traffic class of each packet.

Shimazu wherein each arrival queue is associated with a traffic class, each packet being allocated to the at least one queue in accordance with the traffic class of each packet **(par. 190 teaches each input queue is associated with a priority where each packet is allocated to the queue based on priority).**

See similar motivation as claim 35.

Regarding claim 38, Oner comprising a plurality of transfer queues **(fig. 3, par. 85 teaches a plurality of transfer queues of the packet manager).**

Regarding claim 39, Oner teaches wherein the transfer queue comprises a device level transfer queue and a processor level transfer queue **(par. 85 teaches the two level of transfer queues; fig. 3 teaches device level transfer queues and processor level transfer queues)**, the device level transfer queue configured to receive packets from the arrival queue, and the processor level transfer queue configured to receive packets from the device level transfer queue **(fig. 3 teaches device level transfer queues and processor level transfer queues within element 320 where the device level transfer queue receives packets from the arrival queues 338)**.

Regarding claim 40, Oner teaches wherein packets are transferred to the processor level transfer queue from the device level transfer queue whenever there is space in the processor level transfer queue **(par. 55 teaches places the data in the memory when there is space available).**

Regarding claim 41, Oner teaches wherein packets are never dropped from the transfer queue **(par. 53 teaches transfer queues that buffer data. par. 53 does not teach of dropping packets from the transfer queue)**.

Regarding claim 42, Oner, Shimazu and Cherukuri teach an apparatus as the parent claim.

However Oner and Shimazu do not expressly disclose wherein the processor queues are configured to be associated with different priorities.

Cherukuri teaches wherein the processor queues are associated with different priorities **(col. 7 lines 9-12 teaches the queues are associated with different priorities).**

See similar motivation as claim 35.

Regarding claim 43, Oner teaches wherein the processor is configured responsive to receipt of the interrupt, to remove a packet from a transfer queue, and to classify the packet **(par. 29 teaches transferring a packet for**

processing responsive to an interrupt to a queue and classifies the packet to a queue).

Regarding claim 44, Oner, Shimazu and Cherukuri teach an apparatus as the parent claim.

However Oner and Shimazu wherein the processor is configured to allocate the packet to a processor queue in accordance with a classification of the packet.

Cherukuri teaches wherein the processor is configured to allocate the packet to a processor queue in accordance with a classification of the packet **(col. 7 lines 9-12 suggest packets are allocated to a processor queue in accordance with the priority classification of the packet).**

See similar motivation as claim 35.

Regarding claim 45, Oner teaches wherein the packet is placed in the allocated processor queue if said queue is not full, and otherwise the packet is dropped **(par. 55 teaches places the data in the memory when there is space available).**

Regarding claim 46, Oner teaches a non-transitory computer-readable storage medium encoded with instructions that, when executed on a computer **(par. 20 teaches a memory storing instructions for the computer device)**,

perform a process, the process comprising allocating each received packet to at least one arrival queue **(par. 50 teaches placing the packets into a receive queue; fig. 3 teaches a plurality of arrival queues 338)**, wherein each received packet comprises an internet protocol packet **(par. 27 teaches receiving an IP packet)**;

scheduling, by a scheduler coupled to the at least one arrival queue, packets from the arrival queue to at least one transfer queue **(fig. 3 teaches the scheduler coupled to the at least one arrival queue (receiver buffer - 338); par. 51-53 teaches scheduling, by a scheduler, the packets from the receive buffers to the transfer queues of the packet manager 320; par. 85 further teaches the plurality of data buffers (transfer queues) of the packet manager 320)**;

responsive to transfer of a packet to a transfer queue, generating an interrupt **(par. 77 teaches generating an interrupt for each transfer queue of the packet manager 320)**;

responsive to receipt of an interrupt, allocating the packet from said transfer queue to the processor memory **(par. 58 suggests the packet is sent to the processor memory 340)**;

placing the packet in the processor memory if said processor memory is not full **(par. 55 teaches transferring the data from the transfer queues of the packet manager 320 to the processor (memory 340) when space is available)**, wherein the at least one transfer queue and the processor memory

do not drop packets **(par. 55 suggests packet are not dropped because packets are only transferred as space is available in the memories)** and wherein the scheduler inhibits placement of the packet when the processor memory is full to prevent dropping the packet **(par. 55 teaches packets are only placed in queues when there is space available)**; and

scheduling packets from the processor queues for processing **(par. 55 teaches storing the packets in the memory; par. 58 further teaches synchronizing (scheduling) the data for processing by the CPUs)**, wherein the at least one arrival queue **(receive buffers 338)**, the at least one transfer queue **(transfer queues of packet manager 320)**, and the processor memory **(processor memory 340)** are separate queues **(fig. 3 teaches separate storages)**,

wherein the scheduler includes a first quantity N of inputs each corresponding to the at least one arrival queue, the scheduler further including a second quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N **(fig. 3 teaches a scheduler with at least one input and at least one output, where the input is coupled to the input receive buffers and the output is couple to the transfer queues of the unit 320)**.

Oner suggests in par. 80 that each incoming packet includes various packet attribute values (priority information) in the header of the packet however, Oner does not expressly disclose placing each packet in the allocated arrival

queue if said arrival queue is not full, otherwise dropping said packet; and the processor memory includes a plurality of processor queues.

Shimazu teaches placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet **(par. 190 teaches allocating packets to various queues based on priority; par. 199-200 teaches if the queues are full the packets that are to be placed into each queue will be discarded).**

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner to include prioritizing packets into arrival queues. One would be motivated as such in order to provide different service to the received packets.

However Oner and Shimazu do not expressly disclose the processor memory includes a plurality of processor queues.

Cherukuri teaches a plurality of processor queues **(330-336)** in fig. 3, col. 7 lines 19-22 for receiving packets from transfer queues for future processing.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner and Shimazu to include a plurality of processor queues in the processor memory. One would be motivated as such in order to process packets in order of priority.

6. Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over Oner, Shimazu and Cherukuri in view of Gorti et al. (US Patent No. 6,832,265), hereinafter referred as Gorti.

Regarding claim 13 Oner, Shimazu and Cherukuri teach an apparatus as the parent claim.

However Oner, Shimazu and Cherukuri do not expressly disclose wherein the highest priority queue has the lowest drop probability and the lowest latency.

Gorti teaches wherein the highest priority queue has the lowest drop probability and the lowest latency **(col. 7 lines 18-49 suggests the high priority queue has the lowest drop probability and lowest latency).**

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify the teachings of Oner, Shimazu and Cherukuri to include prioritizing data. One would be motivated as such in order to provide a certain level of service by managing the priority of data packets in the network (abstract).

Conclusion

7. Any response to this action should be **faxed** to (571) 173-8300 or **mailed** to:

Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Hand delivered responses should be brought to:
Customer Service Window
Randolph Building
401 Dulany Street

Alexandria, VA 22314

Any inquiry concerning this communication or earlier communications from the examiner should be directed to DANIEL MITCHELL whose telephone number is (571)270-5307. The examiner can normally be reached on Monday - Friday 8:00 am - 5:00 pm EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Chirag G. Shah can be reached on 571-272-3144. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/D. M./
Examiner, Art Unit 2477

/Gregory B Sefcheck/
Primary Examiner, Art Unit 2477
4/25/2011